온톨로지 의미 매핑 기반 CAD 및 PDM 시스템 정보 통합

이민정*(한국과학기술원 산업공학과), 정원철(한국과학기술원 산업공학과),
이재현(한국과학기술원 산업공학과), 서효원(한국과학기술원 산업공학과)

Ontology Semantic Mapping based Data Integration of CAD and PDM System

Min-Jung Lee(Dept. of Industrial Eng. KAIST), Woncheol Jung(Dept. of Industrial Eng. KAIST),
Jae-Hyun Lee(Dept. of Industrial Eng. KAIST), Hyo-Won Suh(Dept. of Industrial Eng. KAIST)

ABSTRACT

In collaborative environment, it is necessary that the participants in collaboration should share the same understanding about the semantics of terms. For example, they should know that 'Part' and 'Item' are different word-expressions for the same meaning. In this paper, we consider sharing between CAD and PDM data. In order to handle such problems in information sharing, an information system needs to automatically recognize that the terms have the same semantics. Serving this purpose the semantic mapping logic and the ontology based mapper system is described in this paper. In the semantic mapping logic topic, we introduce our logic that consists of four modules: *Character Matching*, *Instance Reasoning*, *definition comparing* and *Similarity Checking*. In the ontology based mapper, we introduce the system architecture and the mapping procedure.

Key Words : Collaboration, Ontology, Semantic mapping, Similarity, CAD, PDM, Integration

1. 서론

CPC 환경은 제품 설계, 엔지니어링, 제조, 구매, 판매, 마케팅 그리고 서비스와 같은 제품의 수명 주기와 관련된 모든 문제들을 통합한 개념이다. CPC 환경에서는 방대한 양의 정보 교환이 발생한다 [1]. 효과적인 협업을 위해서는 제품의 수명 주기와 관련된 모든 참여자들이 다른 참여자들이 의미하는 바와 동일하게 이해할 수 있어야 한다. 그러나 제품의 수명 주기와 관련된 참여자들은 보통 서로 다른 용어를 사용하기 때문에 제품의 수명주기와 관련된 정보시스템을 통합하는데 어려움을 갖게 된다.

모든 도메인에서 사용하는 용어들이 온톨로지적 정의를 갖는다는 가정하에서, 본 연구는 CPC 환경에서도 특히 CAD 정보 의미의 PDM 과의 공유 문제에 대한 해결책으로 온톨로지를 기반으로 한 의미 매핑 로직을 제안한다. 본 로직은 *문자열 비교 (Character Matching)*, *인스턴스 추론 (Instance Reasoning)*, *정의 비교(Definition Comparing)* 그리고 *유사도 측정 (Similarity Checking)*의 4 단계로 구성된다. 그리고 필요에 따라서 정의 비교 단계에서는 세 단계의 매핑 로직이 재귀적으로 사용될 수도 있다. 매핑 로직에 따라서 동일한 의미를 갖는 용어로 판명된 것들은 서로 매핑이 된다. 그러나 그렇지 않은 용어들에 대해서는 사용자가 직접 매핑을 해주어야 한다. 또한 제안된 의미 매핑 로직을 사용하여 CAD 의 정보를 PDM 에서 공유하는 방법을 소개한다. 본 논문에서는 CAD 시스템으로는 SoildEdge 를 사용하였고, PDM 시스템으로는 SmarTeam 을 사용하였다.

본 논문의 구성은 다음과 같다. 2 장에서는 온톨로지의 구조 및 표현언어에 대해서 설명한다. 3 장에서는 의미 매핑 로직을 제안하며, 4 장에서는 제안된 매핑 로직을 바탕으로 하는 CAD 와 PDM 의 정보 공유 방법을 소개한다.

2. 온톨로지 구조 및 표현언어

임의의 도메인을 온톨로지로 표현하기 위해서는

온톨로지의 정의, 구조 그리고 표현 언어의 결정이 선행되어야 한다. 온톨로지의 정의와 구조는 디자이너에 따라 다르다. 본 연구에서는 온톨로지의 기본 요소를 일반적으로 많이 사용되는 5-tuple 구조를 채택하였다[2]. 이 구조의 기본 요소는 다음과 같다.

$$O := \{ C, R, H^C, rel, A^O \}$$

본 연구에서는 온톨로지 표현 언어로서 First Order Logic(FOL)을 선택하였다. 이는 FOL 이 표현 능력이 우수하기 때문이다. Table 1 은 FOL 로 표현된 온톨로지의 예이다.

Table 1 The Example of Ontology Representation by FOL

Element	Example
Concept (C)	*Part(x), endProduct(x)*
Relation (R)	*isSubPartOf(x, y)*
Concept Hierarchy (H^C)	$\forall x$ *endProduct(x)* $\Rightarrow$ *part(x)*
Relation Function (rel)	$\forall$ *x, y isSubPartOf(x, y)* $\Rightarrow$ *part(x)* $\wedge$*part(y)*
Axiom (A^O)	$\forall$ *x, y isSubPartOf(x, y)* $\Rightarrow \neg$ *isSubPartOf(y, x)*

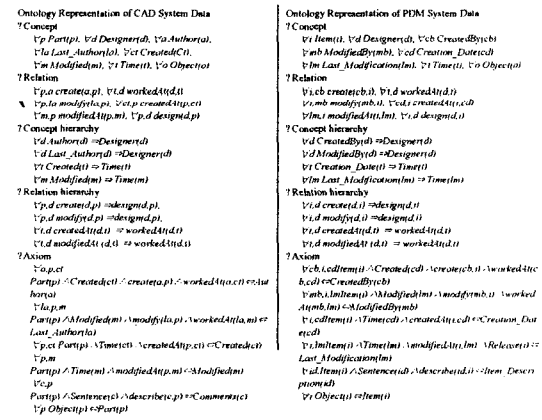

Fig. 1 Ontology Representation of CAD and PDM system data

본 논문에서는 CAD 시스템 (SoildEdge)과 PDM 시스템 (SmarTeam)의 일부 데이터를 표현한 온톨로지를 매핑 로직을 전개하는 예제로 사용한다. Fig. 1 은 CAD 와 PDM 의 일부 데이터의 온톨로지와 인스턴스를 표현한 예이다. Fig. 1 에 의하면 PDM 의 인스턴스 정보가 없음을 알 수 있다. 이는 CAD 에서 PDM 으로 정보가 교환된다면, CAD 에서 사용되는 용어의 인스턴스에 대해서 매핑이 수행 된 이후에야 대응되는 PDM 에서 사용되는 용어의 인스턴스로 변환되기 때문이다.

3. 의미 매핑 로직

본 연구에서는 의미는 같지만 다르게 표현된 용어를 찾는 의미 매핑 로직을 제안하였다. 매핑 로직은 1) *문자열 비교*, 2) *인스턴스 추론*, 3) *정의 비교*, 4) *유사도 측정*의 4 단계로 구성되어있다. 먼저 두 도메인 A 와 B 의 데이터를 얻어서 *문자열 비교*를 통해 용어를 매칭시킨다. 매칭되지 않은 용어들끼리 용어쌍을 만들어 *인스턴스 추론*을 통해 같은 의미를 가지고 있는지 추론한다. 만약에 추론 결과가 *True* 이면, 용어쌍의 각 용어의 하위정의를 정의하여 명확한 결과가 나올 때 까지 전체 로직을 반복적으로 수행하도록 한다. 이를 *정의 비교*라고 한다. 만약에 정의 비교 단계에서 각 도메인의 매칭 안된 하위정의의 용어가 다른 경우에는 *유사도 확인*을 수행한다. 전체 매핑 로직은 Fig. 2 와 같다.

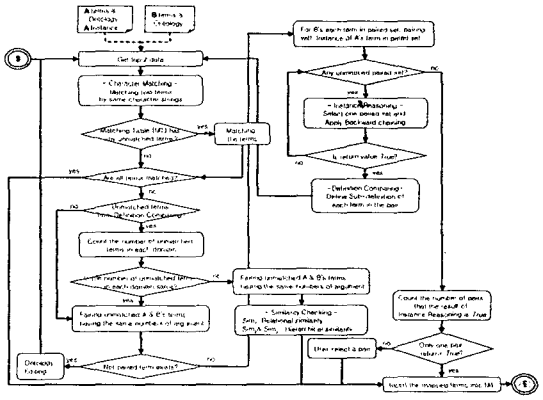

Fig. 2 Semantic Mapping Logic

3.1 문자열 비교 (Character Matching)

이 단계에서는 용어가 동일한 어휘를 갖는다면 동일한 의미를 갖는다고 가정한다. 따라서 두 도메인에서 사용하는 용어 중에서 어휘가 같은 것들끼리는 매핑시킨다. 문자열 비교 절차는 다음과 같다.

1) 두 도메인 A, B 의 온톨로지에서 사용하는 용어의 집합 T_A, T_B 를 각각 정의한다.

$T_A = \{a_1, ?\ \ a_n\}$.

$T_B = \{b_1, ?\ \ b_m\}$.

2) 집합 T_A (T_B)를 집합 C_A (C_B)와 R_A (R_B)로 나눈다.

C_A (C_B): 집합 T_A (T_B)에 있는 컨셉 용어(concept

term)의 집합.

컨셉 용어: *c(i)*와 같이 인자를 1 개 가지는 용어.

R_A (R_B) : 집합 T_A (T_B)에 있는 릴레이션 용어 (relation term)의 집합.

릴레이션 용어: *r(i, j)*와 같이 인자를 2 개 가지는 용어.

3) C_A (R_A)와 C_B (R_B)에 있는 각 용어의 문자열을 비교한다.

4) M_A^C, M_B^C, M_A^R, M_B^R, NM_A^C, NM_B^C, NM_A^R, NM_B^R을 반환한다.

M_A^C (M_B^C): *A* (*B*) 도메인에 있는 컨셉 중 다른 도메인의 어휘와 동일한 어휘를 갖는 컨셉의 집합.

M_A^R (M_B^R): *A* (*B*) 도메인에 있는 릴레이션 중 다른 도메인의 어휘와 동일한 어휘를 갖는 릴레이션의 집합.

NM_A^C (NM_B^C): *A* (*B*) 도메인에 있는 컨셉 중 다른 도메인의 어휘와 동일한 어휘가 없는 컨셉의 집합.

NM_A^R (NM_B^R): *A* (*B*) 도메인에 있는 릴레이션 중 다른 도메인의 어휘와 동일한 어휘가 없는 릴레이션의 집합.

CAD 의 정보와 PDM 의 정보를 표현하는데 사용되는 용어가 각각 Fig. 1 과 같다면, Fig. 3 과 같은 결과를 얻을 수 있다.

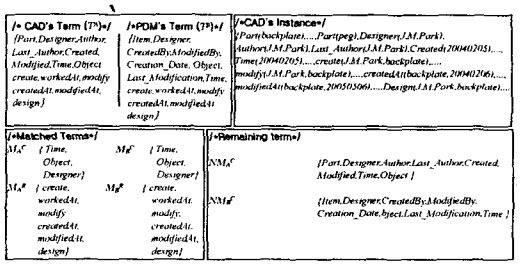

Fig. 3 The Example of *Character Matching*

3.2 인스턴스 추론 (Instance Reasoning)

이 단계에서는 위에서 매칭 안된 용어들의 쌍을 만들어 각 용어쌍이 같은 의미를 가지는지 추론한다. 본 추론은 각 용어쌍의 A 도메인의 용어의 인스턴스가 B 도메인의 용어 정의를 만족하는지 Backward Chaining Algorithm 을 이용하여 평가한다 [3]. 자세한 절차는 다음과 같다.

1) NM_A^C (NM_A^R)와 NM_B^C (NM_B^R)에 있는 컨셉 (릴레이션) 용어들의 쌍을 만든다.

2) 용어쌍들을 A 도메인의 용어로 a_i 를 갖는 컨셉(릴레이션) 용어쌍들의 집합 Sa_i^C (Sa_i^R) 을 만든다.

3) 각 용어쌍의 A 도메인의 컨셉 (릴레이션) 용어 a_i^C (a_i^R)의 인스턴스가 B 도메인의 컨셉(릴레이션) 용어 b_j^C (b_j^R)의 정의를 만족하는지 확인한다.

4) a_i^C (a_i^R)의 인스턴스가 b_j^C (b_j^R)의 정의를 만족하면 *True* 와 집합 TSa_i 를 반환하고, 정의 비교 단계를 수행한다.

그렇지 않으면, 남은 용어쌍이 있는 경우 5) 단계를 수행하고, 없으면 의미 매핑 로직 절차를 종료한다.

TSa_i : A 도메인의 용어로 a_i 를 갖는 용어쌍 중에서 A 도메인 용어의 인스턴스가 B 도메인의 용어 정의를 만족하는 쌍의 집합.

앞의 예로부터 Fig. 4 를 이용하여, Fig. 5 와 같은 결과를 얻을 수 있다.

Fig. 4 Input Information for *Instance Reasoning*

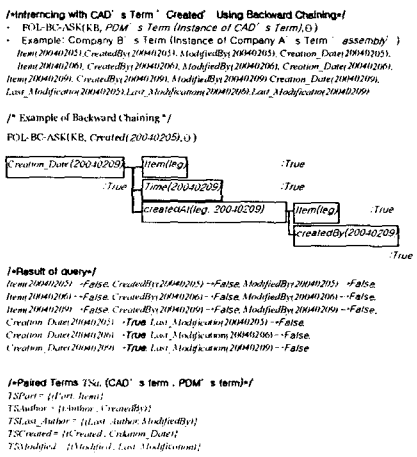

Fig. 5 The Result of *Instance Reasoning*

3.3 정의 비교 (Definition Comparing)

이 단계에서는 *인스턴스 추론* 단계의 결과로 나온 각 용어쌍에 다음과 같은 단계를 수행한다.

1) 집합 SD_{a_i}와 SD_{b_j}를 정의한다.

SD_{a_i} : TS_{a_i}에 있는 A 도메인의 용어 a_i^C (a_i^R)의 하위 정의들의 집합.

SD_{b_j} : TS_{a_i}에 있는 B 도메인의 용어 b_j^C (b_j^R)의 하위 정의들의 집합.

2) SD_{a_i} (SD_{b_j})를 T_A (T_B)로 둔다.

3) T_A (T_B)에 의미 추론 로직 전체를 적용한다.

두 용어가 동일한 의미를 갖는지에 대한 구체적인 결과(*True/False*)를 얻을 때까지 로직을 재귀적으로 반복한다. 본 연구에서는 이에 대해서 다음과 같은 두 가지 가정을 갖고 있다. (1) 루프에 빠지는 경우는 없다. (2) 하위정의들을 로직에 적용하는 순서는 정의에 나타난 순서대로 한다.

4) 정의가 같은 용어는 집합 CS_{a_i}에 삽입한다.

CS_{a_i}: A 도메인의 용어 a_i^C (a_i^R)와 동일한 정의를 갖는 것으로 결론지어진 용어쌍들의 집합.

5) $n = 0$ 인 경우, TS_{a_i}에 있는 모든 용어쌍에 유사도 확인 단계를 수행한다.

$n = 1$ 인 경우, 집합 MT_{a_i}에 용어쌍을 삽입한다.

$n \geq 2$ 인 경우, 사용자가 용어쌍을 선택해서 집합 $MT_{a_{ii}}$에 삽입한다.

MT_{a_i}: a_i^C (a_i^R)가 포함된 매핑된 용어쌍.

n : 집합 CS_{a_i}에 들어있는 용어쌍의 수.

Fig. 6 은 *정의 비교*의 예를 보여주고 있다.

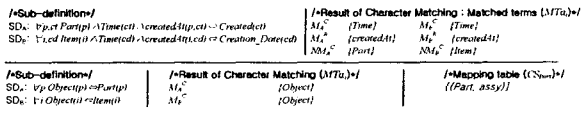

Fig. 6 The Result of *Definition Comparing* with '*Created*?and *Creation_Date*'

3.4 유사도 정의 (Similarity Checking)

본 연구에서는 Fig.7 과 같은 온톨로지 구조를 기반으로 하는 두 가지 종류의 컨셉 유사도를 고려한다. 하나는 *관계성 유사도 (Relational Similarity)*이고 나머지 하나는 *계층적 유사도 (Hierarchical Similarity)* 이다. 이는 릴레이션 용어가 아니라 컨셉 용어의 유사도만 고려한다. 먼저 두 가지 유사도를 구해서 두 값의 가중합을 계산한다.

3.4.1 관계성 유사도 (Relational similarity)

본 유사도는 근처의 컨셉들이 서로 매핑이 되었다면, 이들 컨셉들도 서로 매핑이 될 가능성이 높다 [4]는 개념을 이용하였다. Fig. 7 에 기반으로 하는 절차는 다음과 같다.

1) 두 집합 Ra_i와 Rb_j 를 정의한다.

Ra_i (Rb_j): 용어 a_i^C (b_j^C)를 도메인 혹은 레인지로 갖는 릴레이션 용어의 집합.

2) Ra_i에 있는 릴레이션 용어와 Rb_j에 있는 릴레이션 용어 중에서 동일한 어휘를 갖고, 도메인에 해당하는 용어 혹은 레인지에 해당하는 용어가 같은 릴레이션 용어들의 쌍을 원소를 갖는 집합 SR_{a_i, b_j}를 정의한다.

3) a_i^C 와 b_j^C 간의 관계성 유사도 $Sim_1(a_i^C, b_j^C)$는 다음과 같이 구한다.

$Sim_1(a_i^C, b_j^C) = 0$, if p=0 or r=0.

$Sim_1(a_i^C, b_j^C) = \alpha$ (사용자 정의), if p=0 and r=0.

$Sim_1(a_i^C, b_j^C) = k / p$.

p: 집합 R_{ai}에 들어있는 릴레이션의 수.

r : 집합 Rb_j에 들어있는 릴레이션의 수.

k : 집합 SR_{a_i, b_j}에 들어있는 용어쌍의 수.

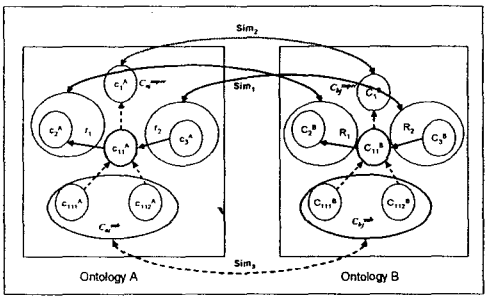

Fig. 7 Similarity Checking

3.4.2 계층적 유사도 (Hierarchical similarity)

본 유사도는 두 컨셉의 부모 컨셉 혹은 자식 컨셉 간에 서로 매핑이 되어있다면, 이들 간에도 매핑이 될 가능성이 높다는 개념을 이용하였다. 계층적 유사도는 직접적인 상위 컨셉간과 직접적인 하위 컨셉간의 유사도를 뜻한다. Fig. 7 에 기반으로 하는 절차는 다음과 같다.

1) 집합 Ca_i^{super}, Ca_i^{sub}, Cb_j^{super}, Cb_j^{sub} 을 정의한다.

Ca_i^{super} (Cb_j^{super}): a_i (b_j)의 직접적인 상위 컨셉의 집합.

Ca_i^{sub} (Cb_j^{sub}): a_i (b_j)의 직접적인 하위 컨셉의 집합.

2) Ca_i^{super} (Ca_i^{sub})에 해당하는 용어 중에서 Cb_j^{super}(Cb_j^{sub})에도 동시에 속하는 용어, 즉 서로 일치하는 용어들의 쌍을 갖는 집합 $SHR_{a_i, b_j^{super}}$($SHR_{a_i, b_j^{sub}}$)을 정의한다.

3) a_i^C 와 b_j^C 간의 상위 컨셉과 관련된 계층적 유사도 $Sim_2(a_i^C, b_j^C)$는 다음과 같이 구한다.

$Sim_2(a_i^C, b_j^C) = 0$, if p=0 or r=0.

$Sim_2(a_i^C, b_j^C) = \alpha$ (사용자 정의), if p=0 and r=0.

$Sim_2(a_i^C, b_j^C) = m / p$.

p: 집합 Ca_i^{super}에 들어있는 릴레이션 용어의 수.

r: 집합 Cb_j^{super}에 들어있는 릴레이션 용어의 수.

m: 집합 SHR_{a_i, b_j}^{super}에 들어있는 용어쌍의 수.

4) a_i^C 와 b_j^C 간의 하위 컨셉과 관련된 계층적 유사도 $Sim_3(a_i^C, b_j^C)$는 다음과 같이 구한다.

$Sim_3(a_i^C, b_j^C) = 0$, if p=0 or r=0.

$Sim_3(a_i^C, b_j^C) = \alpha$ (사용자 정의), if p=0 and r=0.

$Sim_3(a_i^C, b_j^C) = u / p$.

p: 집합 Ca_i^{sub}에 들어있는 릴레이션 용어의 수.

r: 집합 Cb_j^{sub}에 들어있는 릴레이션 용어의 수.

u: 집합 SHR_{a_i, b_j}^{sub}에 들어있는 용어쌍의 수.

3.4.3 총 유사도 (Total similarity)

최종적인 두 용어간의 유사도는 관계성 유사도와 계층적 유사도를 이용하여 아래와 같은 방법으로 얻어진다. 각 유사도에 대한 가중치는 사용자의 판단에 따라서 값을 부여하게 된다.

1) $Sim(a_i, b_j)$ 을 계산한다.

$Sim(a_i, b_j) = a_1 Sim_1(a_i, b_j) + a_2 Sim_2(a_i, b_j) + a_3 Sim_3(a_i, b_j)$

$a_1 + a_2 + a_3 = 1$, $a_1 \geq 0$, $a_2 \geq 0$, $a_3 \geq 0$

2) 유사도와 함께 매핑 가능한 용어들이 사용자에게 제시되면, 사용자는 매핑 여부를 결정한다.

앞의 예제에서 위의 *유사도 확인*을 수행한 결과는 Fig. 8 에 나타나 있고, 전체 의미 매핑 로직을 통해서 매핑된 결과는 Fig. 9 와 같다.

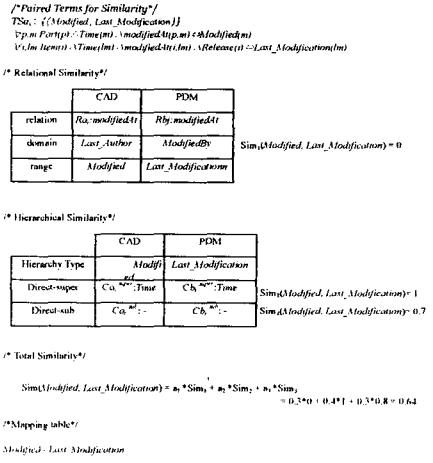

Fig. 8 The Example of *Similarity Checking*

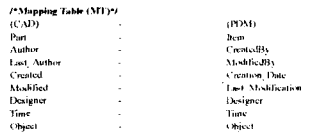

Fig. 9 The Result of Example for Semantic Mapping

4. Semantic Ontology based Mapper (SOM)

4.1 SOM 개발 환경

SOM 시스템은 크게 네 부분으로 분류 된다. 우선, Core Mapper 는 본 연구에서 제안된 매핑 로직이 구현되어 있는 부분이며, C# 언어로 구현하였다. Ontology Translator 는 XML 을 Prolog 로 혹은 그 반대로 변환 시키는 기능을 수행하며, 마찬가지로 C# 언어를 이용하였다. 다음으로, Ontology Base 는 용어에 대한 온톨로지를 갖고 있으며, 매핑 결과가 저장되는 저장소를 의미한다. 이 부분은 MS-SQL 서버로 대체하였다. 마지막으로 Inference Module 은 매핑 수행 중 역방향 추론을 이용해야 하는 경우가 발생하는데, 그 기능을 대신 수행하는 부분이다. 질의에 대해서 간단히 Yes/No 의 결과를 반환하는 Prolog 를 선택하였으며, Prolog 와 C# 언어간에 직접적인 인터페이스가 제공되지 않기 때문에, JAVA 언어로 인터페이스 역할을 수행하는 부분을 구현하였다.

4.2 SOM 프로시져

SOM 시스템은 CAD 용어로 표기된 XML 형식의 문서를 읽어 들여서, 매핑된 PDM 용어로 표기된 XML 형식의 문서를 작성한다. 다음과 같은 하위 과정들로 나눌 수 있다.

Fig. 10 은 SoildEdge 로 그린 의자와 의자의 각 파트의 정보이다. 이 정보는 Text file 형식으로 얻을 수 있으며 이를 XML 로 변환하여 SOM 시스템에서 이용한다.

선행 조건: 입/출력 정보로 사용되는 XML 문서의 정합성을 검증할 수 있도록 XML 스키마가 주어져야 한다. 또한 두 시스템 용어의 온톨로지가 미리 Ontology base 에 정의되어 있어야 한다.

입력 문서 처리: SOM 시스템은 인스턴스 정보를 갖는 XML 형식의 입력 문서를 XML 스키마를 참조하여 검증한다. 그리고 SOM 시스템은 입력 문서의 정보를 미리 정의된 온톨로지를 이용하여 검증을 하며, 이 과정을 마치면 Prolog 형식의 파일이 작성된다. 이 파일은 사용자에 의해서 컴파일 되고 저장된다.

매핑 수행: Prolog 파일과 두 회사의 용어에 대한 온톨로지 정의를 이용하여, 매핑 로직에 의해 용어 매핑이 수행된다. 두 회사 용어간의 매핑 정

보는 Mapping table 에 저장된다.

출력 문서 작성: 매핑 정보를 이용하여, CAD 의 인스턴스 정보를 PDM 용어를 사용하여 표현할 수 있다. 우선, 온톨로지 정합성 검증을 거쳐서 Prolog 형식의 출력 문서를 작성하게 된다. 이 문서는 PDM 의 XML 스키마를 참조하여 정합성을 검증한다. 검증이 끝나면 PDM 의 용어로 표현된 XML 형식의 문서가 최종적으로 작성된다.

Fig. 10 은 CAD 시스템 (SoildEdge)의 정보이고, Fig. 11 은 CAD 의 정보가 공유된 PDM 시스템 (SmarTeam)과 정보의 XML 형식의 문서이다.

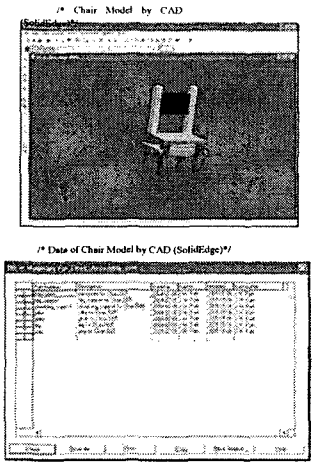

Fig. 10 CAD System (SoildEdge) and Data

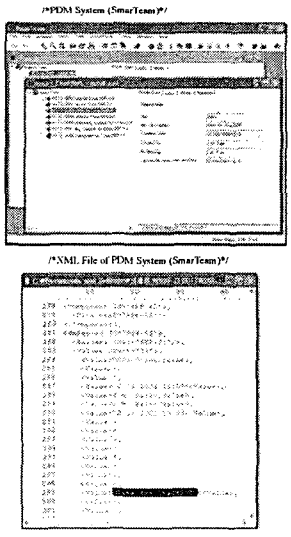

Fig. 11 PDM System (SmarTeam) and XML file

5. 결론

본 연구에서는 서로 다른 용어를 매핑 시키는 의미 매핑 로직을 제안하였으며, Semantic Ontology Mapper (SOM) 시스템을 개발하였다. 매핑 로직은 문자열 비교, 추론, 정의 비교, 유사도 검사의 4 단계로 이루어진다. SOM 시스템은 용어 매핑 결과를 바탕으로 입력 문서가 갖고 있는 A 도메인의 정보를 B 도메인의 용어로 변환시켜준다. 따라서, SOM 시스템은 특별한 표준에 따르지 않고도, 이기종 시스템간 정보 공유 가능하게 한다. 특히 이 논문에서는 CAD 시스템 (SolidEdge) 의 일부 정보를 PDM 시스템 (SmarTeam)과 공유할 수 있는 방법을 제안하였다.

제안된 의미 매핑 로직의 제한점은 다음과 같다. CPC 환경에서의 자유로운 정보공유를 위해서는 매핑 알고리즘에 대한 추가적인 연구가 필요하다. 현재의 로직은 의미가 다르고, 표현이 다른 용어에 관해서는 사용자의 판단에 맡기고 있다. 따라서 일반적인 경우에 대한 로직의 개발이 유연한 정보 공유를 위해서 필요하다.

매핑 기술의 적용을 위해서는 각 기업이 온톨로지를 구축해야 한다. 그러나 온톨로지 디자인은 방대한 양의 지식공학 작업, 많은 시간과 노력을 필요로 한다. 따라서 이에 대한 간편하고 효율적인 접근법이 개발되어야 할 것이다.

감사의 글

본 연구는 산업자원부에서 추진하는 차세대신기술개발사업의 하나로 수행되고 있는 '글로벌 정보 공유 및 지식기반의 차세대 생산시스템 개발' 과제의 지원을 받아 수행되었습니다.

참고문헌

1. Aberdeen Group, "Collaborative Product Commerce: Delivering Product Innovations at Internet Speed," Market view point Vol. 12, No. 9, 1999.
2. A. Meadche, "Ontology Learning for the Semantic Web," Kluwer Academic Publisher, pp14-19, 2002.
3. A.D. Preece, K. J. Hui, W.A. Gray, P. Marti, T.J.M. Bench-Capon, D.M. Jones, and Z. Cui, "The kraft architecture for knowledge fusion and transformation," Proceedings of the 19th SGES International Conference on Knowledge-Based Systems and Applied Artificial Intelligence (ES? 9). Springer, pp23-40, 1999.
4. W. Li, and C. Clifton, "SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks," Data & Knowledge Engineering, pp49-84, 2000.